Spatial Data Standard
for facilities, infrastructure, & environment
(SDSFIE)

and

Facility Management Standard
for facilities, infrastructure, & environment
(FMSFIE)

Compliance Policy

The CADD/GIS Technology Center
For facilities, infrastructure, and environment
Information Technology Laboratory
U.S. Army Engineer Research and Development Center
Vicksburg, Mississippi

Foreword:

Since 1993, the Spatial Data Standard for facilities, infrastructure, & environment (SDSFIE) (formerly called TSSDS & SDS) and Facility Management Standard for facilities, infrastructure, & environment (FMSFIE) (formerly called TSFMS & FMS) have continued to expand to meet the needs of an increasing customer community. From Release 1.600 to present, the number of "Features" (Entity Types) have grown from just over 600 to more than 1000 and the number of "Attributes" have increased from less than 10,000 to nearly 26,000. For Release 2.000, an unpopulated database created from the entire SDSFIE/FMSFIE schema will require approximately 75 megabytes (MB) of hard drive space.

Due to it's generic nature, many SDSFIE/FMSFIE tables contain a large number of attributes. Many customers have indicated that they prefer to only maintain the SDSFIE/FMSFIE attributes which they need, and prefer to be able include their "local" attributes (i.e., the attributes not currently included in the SDSFIE/FMSFIE) within the SDSFIE/FMSFIE attribute tables.

In 1997, the concept of a "Filter" was introduced to provide customers the capability to build and manage less than the full SDSFIE/FMSFIE schema (i.e., the entire SDSFIE/FMSFIE data model/dataset). The "Filter" capability (as provided by the "Filter Maker" software application) permits customers to select subsets of the SDSFIE/FMSFIE schema for their specific functions & projects.

Up to the present time, the policy for "SDSFIE/FMSFIE compliance" has dictated that: "if a SDSFIE/FMSFIE attribute table (i.e., an attribute table included in the SDSFIE/FMSFIE schema) was present in the database, then all SDSFIE/FMSFIE attributes (i.e., attributes included in each specific SDSFIE/FMSFIE attribute table) must be present in that table. In addition, all attributes in the table must use the formatting, ordering, and naming conventions of the SDSFIE/FMSFIE. Also, no customer defined attributes have been permitted to be added to a SDSFIE/FMSFIE table."

The SDSFIE/FMSFIE "SQL Generator", "Access Builder", & "GeoMedia Builder" tools are designed to build, test, and upgrade databases that meet the "Basic Level" of SDSFIE/FMSFIE compliance. The "Basic" level of SDSFIE/FMSFIE compliance would be recommended: (1) For initial GIS implementations, (2) For customers who do not have experienced database administrators and GIS analysts, and (3) Where "local" policies and standards have not yet been developed for ensuring quality and integrity of data collected and delivered by multiple sources.

The "Experienced Level"of SDSFIE/FMSFIE compliance would be recommended: (1) For customers who have experienced database administrators (DBA's) and/or GIS analysts, and (2) Where "local" policies and standards have been developed for ensuring quality and integrity of data collected and delivered by multiple sources.

The SDSFIE and FMSFIE are designed to be used with an external relational database (i.e. Oracle, Informix, Access, etc.) rather than the flat file data structure (e.g., ESRI coverage & shape files, and CADD objects). The use of an external relational database can provide many benefits, such as: (1) provide the ability to support multiple users accessing the data simultaneously using different CADD, GIS, and other software products, (2) provide an enterprise GIS solution where data maintained in one database format (can be maintained by one office or maintained in multiple offices) which is available to all users at an installation or within an organization; (3) provide a non-proprietary format with greater flexibility to share data with other users and applications; (4) provide a stable data format which protect an organization's data investment (vendors may discontinue products and/or support of a proprietary data structure at some time in the future); (5) avoid possible loss of data when upgrading to newer versions of CADD/GIS software, or exporting graphic and non-graphic data to other CADD/GIS software products/applications; and (6) maximize the return on investment by providing the capability to use data in an organization's business functions (e.g., generating reports, statistical analyses, etc.) using database, programming, & web tools. However, if GIS users do not yet have the experience or resources to establish and maintain an external relational database for GIS, they should use the SDSFIE and FMSFIE as data dictionaries (i.e., use SDSFIE/FMSFIE attribute & domain value naming conventions, data types, etc.). Using the SDSFIE and FMSFIE as data dictionaries for flat file data structures can simplify and save costs on future conversion of the flat files to a SDSFIE/FMSFIE compliant relational database.

"Basic Level"

(Supported by SDSFIE/FMSFIE "SQL Generator", "Access Builder", & "GeoMedia Builder" tools)

1. *Entity Types & Entities* - Use SDSFIE Entity Type and Entity naming conventions (i.e., the naming conventions included in the SDSFIE). The SDSFIE symbology (i.e., colors, line styles, and symbols) is recommended, and customers have the option of using their own symbology.

2. *Attribute Tables* - Use SDSFIE/FMSFIE attribute table naming conventions and definitions (short names (8 characters in length), the SQL Generator and Access Builder software applications currently constructs attribute tables using the SDSFIE/FMSFIE short names). It is not a requirement to use the entire SDSFIE/FMSFIE schema. A "subset" of the SDSFIE/FMSFIE schema may be used by: (1) Selecting one of the custom "Filters" provided with each SDSFIE/FMSFIE release, or (2) Using the "SDSFIE/FMSFIE Filter Maker" software application to build a custom filter based on the specific Entity Types required. The "Filters" will provide a SDSFIE/FMSFIE schema containing all of the selected "graphic" attribute tables (i.e., those attribute tables associated with the selected Entity Types) as well as all linked/joined SDSFIE/FMSFIE attribute tables. The unneeded linked/joined SDSFIE/FMSFIE attribute tables may be deleted.

3. *Attributes* - Use defined SDSFIE/FMSFIE Attribute naming conventions and definitions (short names (10 characters in length), the SQL Generator and Access Builder software applications currently construct tables using the SDSFIE/FMSFIE short names). All SDSFIE/FMSFIE defined attributes, as output by the SDSFIE/FMSFIE software tools (e.g., SQL Generator and Access Builder), must be kept for the selected tables. Some options are being built into the software tools, which permit Users to exclude specific pre-defined attributes.

If a locally configured attribute is required, determine the SDSFIE/FMSFIE table to which the attribute is associated with. Determine the Primary Key of that table. This information is easily obtained from the SDSFIE/FMSFIE Browser software application. In the database, create a new table. Name this table using the SDSFIE/FMSFIE table name, followed by a locally assigned prefix. As an example, “local_” would work quite well. Thus, if an attribute is to be added to the “bggenstr” SDSFIE/FMSFIE attribute table, a table “local_bggenstr” table would be created. The very first attribute in the table would be “buildng_id”, the very same attribute that is the Primary Key of the bggenstr table.

The following figure shows a graphical depiction of the arrangement being recommended. The two tables parallel each other inside the users database.

This structure and naming convention has several advantages. First, all of the "local" tables will appear together in a tables list, and they are easily correlated to the corresponding SDSFIE/FMSFIE attribute table. This guarantees that the "local" attribute table will not be confused with a SDSFIE/FMSFIE attribute table, since the name exceeds the 8 character SDSFIE/FMSFIE limit and begins with the defined local identifier. By naming the first attribute as the Primary Key of the corresponding SDSFIE/FMSFIE table, it is possible to create a join on the "buildng_id" in bggenstr with "buildng_id" in local_bggenstr which provides the correct instance correlation.

When, and if, the CADD/GIS Technology Center adds an attribute to the SDSFIE/FMSFIE which corresponds to the locally configured attribute, it is possible to run a single SQL query which will update the values in the SDSFIE/FMSFIE table from the local table. That SQL statement would look something like:

UPDATE bggenstr INNER JOIN local_bggenstr ON bggenstr.buildng_id = local_bggenstr.buildng_id SET bggenstr.gate_num = local_bggenstr.gate_num;

The local attribute can then be deleted from the "local" attribute table, or the local attribute table may be deleted if that is the only local attribute in the table.

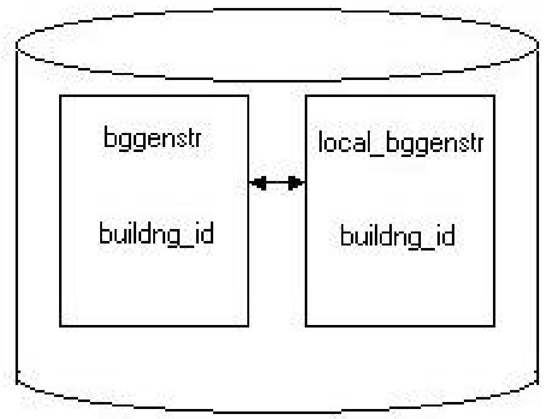


Figure - Creating a Local Table

4. *Domain Values* - Use SDSFIE/FMSFIE domain table naming conventions and definitions (short names (8 characters in length), the SQL Generator and Access Builder software applications currently constructs domain tables using the SDSFIE/FMSFIE short names (i.e., Domain Table Name)). It is not a requirement to use the entire SDSFIE/FMSFIE schema. A "subset" of the SDSFIE/FMSFIE schema may be used by: (1) Selecting one of the custom "Filters" provided with each SDSFIE/FMSFIE release, or (2) Using the

"SDSFIE/FMSFIE Filter Maker" software application to build a custom filter based on the specific Entity Types required. The "Filters" will provide a SDSFIE/FMSFIE schema containing all of the selected "graphic" attribute tables (i.e., those attribute tables associated with the selected Entity Types) as well as all linked/joined SDSFIE/FMSFIE attribute tables. All of the SDSFIE/FMSFIE domain tables associated with the selected and linked/joined attribute tables are also provided. The unneeded linked/joined SDSFIE/FMSFIE attribute tables and associated domain tables may be deleted.

5. *List and Range Domain Values* - Use defined SDSFIE/FMSFIE List and Range Domain Value naming conventions and definitions (short names (16 characters in length) the SQL Generator and Access Builder software applications currently construct domain tables using the SDSFIE/FMSFIE domain value short names). All SDSFIE/FMSFIE defined list and range domain values must be kept for the selected domain tables. "Local" list and range domain tables may be included in the database using the same procedures as defined for "local" attribute tables.

"Experienced Level"

(Center currently does not provide tools which support customer customization of tables (i.e., deleting and adding attributes to SDSFIE/FMSFIE tables).

1. *Entity Types & Entities* - Use SDSFIE Entity Type and Entity naming conventions (i.e., the naming conventions included in the SDSFIE). The SDSFIE symbology (i.e., colors, line styles, and symbols) is recommended, and customers have the option of using their own symbology.

2. *Attribute Tables* - Use SDSFIE/FMSFIE attribute table naming conventions and definitions (either short (8 characters in length) or common (long, up to 50 characters in length) are acceptable). It is not a requirement to use the entire SDSFIE/FMSFIE schema. A "subset" of the SDSFIE/FMSFIE schema may be used by: (1) Selecting one of the custom "Filters" provided with each SDSFIE/FMSFIE release, or (2) Using the "SDSFIE/FMSFIE Filter Maker" software application to build a custom filter based on the specific Entity Types required. The "Filters" will provide a SDSFIE/FMSFIE schema containing all of the selected "graphic" attribute tables (i.e., those attribute tables associated with the selected Entity Types) as well as all linked/joined SDSFIE/FMSFIE attribute tables. The unneeded linked/joined SDSFIE/FMSFIE attribute tables may be deleted.

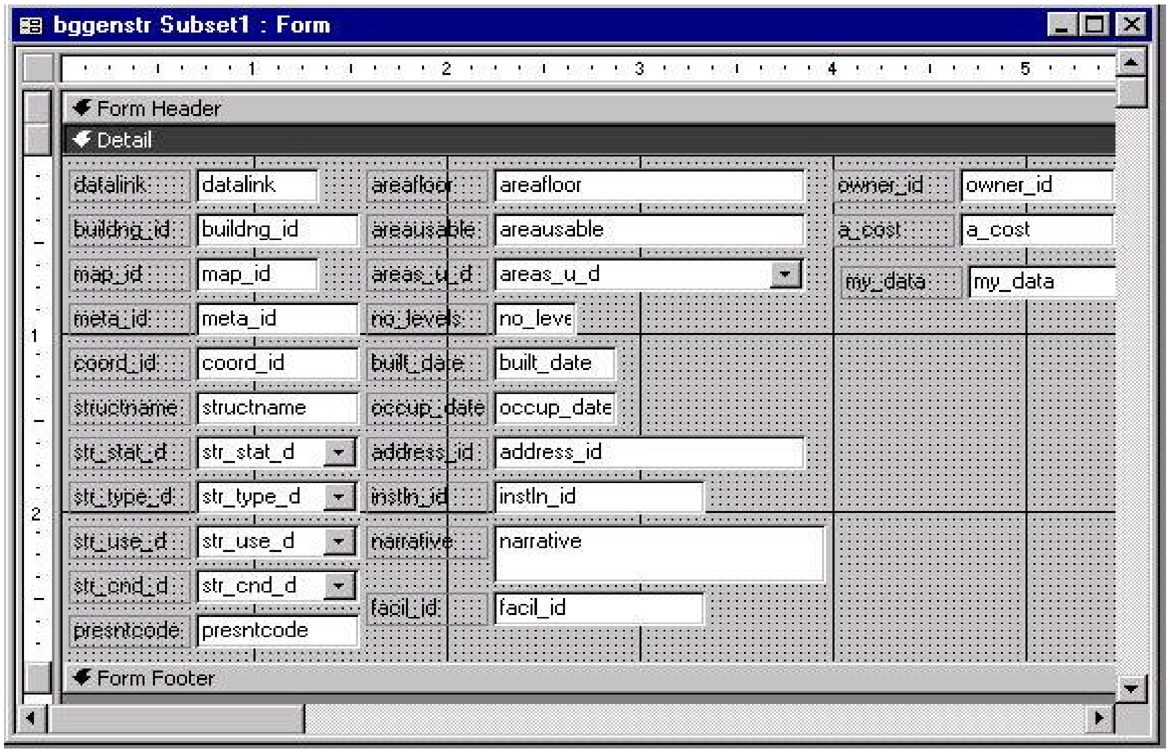
3. *Attributes* - Customers have the flexibility of adding "local" attributes, and deleting SDSFIE/FMSFIE attributes not required for the customer's GIS/FM data collection, analysis, and reporting requirements. SDSFIE/FMSFIE formatting, ordering, and naming conventions must be maintained for the SDSFIE/FMSFIE attributes retained in each attribute table. The customer cannot add "local" attributes which duplicate the SDSFIE/FMSFIE attributes. For the SDSFIE/FMSFIE attributes, either the short name (10 characters in length) or long name (common name, up to 50 characters in length) may be used.

The following attributes must be retained in all SDSFIE and FMSFIE attribute tables:

a. Primary Key - The "primary key" attribute is required for assigning a unique identifier to the data record and for building the relationships (joins) with similar records used by the relational database software. The attribute entitled "buildng_id" is the primary key for the table "bggenstr" (see illustration below).

The illustration below entitled "bggenstr Subset1" provides an example of a possible "Experienced" Level SDSFIE/FMSFIE implementation. The SDSFIE table entitled "bggenstr" contains a total of 62 attributes. The customer has determined that they only need to collect and maintain data for 23 of the

SDSFIE/FMSFIE attributes, and they have added one "local" attribute entitled "my_data".



datalink:	datalink	areafloor:	areafloor	owner_id:	owner_id
buildng_id:	buildng_id	areausable:	areausable	a_cost:	a_cost
map_id:	map_id	areas_u_d:	areas_u_d	my_data:	my_data
meta_id:	meta_id	no_levels:	no_leve		
coord_id:	coord_id	built_date:	built_date		
structname:	structname	occup_date:	occup_date		
str_stat_d:	str_stat_d	address_id:	address_id		
str_type_d:	str_type_d	instln_id:	instln_id		
str_use_d:	str_use_d	narrative:	narrative		
str_cnd_d:	str_cnd_d	facil_id:	facil_id		
presntcode:	presntcode				

“Experienced Level” Users also have the following options:

a. The order of the attributes in a table may be changed.

b. “Date/Time” data types may be used in lieu of the SDSFIE/FMSFIE integer date/time fields.

c. “Boolean” (Yes/No) data types may be used in lieu of the SDSFIE/FMSFIE Boolean Domain Table (d_boolean).

d. Shorter character fields than those specified in the SDSFIE/FMSFIE may be used.

4. *Domain Tables* - Use defined SDSFIE/FMSFIE domain table naming conventions and definitions (either short (Table Name, 10 characters in length) or long (Domain Name, up to 32 characters in length) names are acceptable). It is not a requirement to use the entire SDSFIE/FMSFIE data structure. A "subset" of the SDSFIE/FMSFIE schema may be used by: (1) Selecting one of the custom "Filters" provided with each SDSFIE/FMSFIE release, or (2) Using the "SDSFIE/FMSFIE Filter Maker" software application to build a custom filter based on the specific Entity Types required.

5. *Domain Values* - Customers have the flexibility of adding "local" domain values, and deleting SDSFIE/FMSFIE domain values not required for the customer's GIS/FM data collection, analysis, and reporting requirements.

SDSFIE/FMSFIE formatting, ordering, and naming conventions must be maintained for the SDSFIE/FMSFIE domain values retained. The customer cannot add local domain values that are a duplicate of those defined in the SDSFIE/FMSFIE domain tables. For SDSFIE/FMSFIE List Domain Values, the either the short name (Value, up to 16 characters in length) or long name (Full Name, up to 32 characters in length) may be used.

Definitions

Entity Set. *Entity Sets* are the highest level of the SDSFIE data model structure and represent data organized at the project level. *Entity Sets* are broad, generalized themes containing groupings (called Entity Classes) of features (i.e., graphic objects (called Entity Types) which can be depicted at their actual geographic locations on a map) and related "graphic" attribute data (i.e., data (information) about the feature which is stored in a database table). The SDSFIE Release 2.00 structure contains the following twenty-six Entity Sets: (1) Auditory, (2) Boundary, (3) Buildings, (4) Cadastre, (5) Climate, (6) Common, (7) Communications, (8) Cultural, (9) Demographics, (10) Environmental Hazards, (11) Ecology, (12) Fauna, (13) Flora, (14) Future Projects, (15) Geodesy, (16) Geology, (17) Hydrography, (18) Improvements, (19) Landform, (20) Land Status, (21) Military Operations, (22) Olfactory, (23) Soil, (24) Transportation, (25) Utilities, (26) and Visual.

The appropriate *Entity Set* name is reflected in the first two characters of each attribute table name code.

For CADD (e.g., MicroStation and AutoCAD) and CADD-based GIS (e.g., MGE, AutoDesk Map, and Bentley GeoGraphics), the Entity Set name is also represented in the first two characters of each design file name code.

Entity Class. *Entity Classes* comprise the next level of the hierarchical SDSFIE data model structure. *Entity Classes* contain groupings of similar features (called Entity Types) and related "graphic" attribute data. Each Entity Class is equivalent to a separate map or drawing file. Equivalent names used by various CADD and GIS software vendors are provided in the following table:

CADD/GIS Software (Vendor)	**Counterpart Name for Entity Class**
MGE (Intergraph)	Category or Design file
ARCINFO (ESRI)	Workspace
MicroStation (Bentley)	Design file
AutoCAD (AutoDesk)	Drawing File

The name of *Entity Class* is represented by a three character code which makes up a part of the attribute table name codes (and design/drawing files name codes for CADD and CADD-based GIS).

Entity Type. Each *Entity Class* contains one or more *Entity Types*. An *Entity Type* is the logical name assigned to a graphic feature (i.e., an object that can be graphically depicted on a map or drawing). Each *Entity Type* has a corresponding "graphic" attribute table containing specific information about the *Entity Type*.

An *Entity Type* is equivalent to a "coverage" in ArcInfo and a "view" in ESRI ArcView. *Entity Types* in MGE are grouped by features.

For CADD (e.g., MicroStation and AutoCAD) and CADD-based GIS (e.g., AutoDesk Map and Bentley GeoGraphics) software, *Entity Types* represent a grouping of like cartographic (or CADD) elements (called *Entities*) assigned to separate levels/layers.

*Discriminators***.** Effective use of GIS relies on the ability of the user to adequately differentiate subtle differences in geographical features, or *Entities*. This differentiation permits greater value in output products by selectively displaying *Entities* based on some pre-defined criteria. While some differentiation is determined by the assignment of the graphical properties (color, for example), it is often useful to expand the capability of this differentiation to only display selected *Entities* (i.e., displaying a map of only the paved roads).

Historically, the CADD user has accomplished this differentiation by assigning these different *Entities* to different layers or levels within the drawing file. Virtually all CADD systems provide a simple capability to turn various layers on or off to allow for the display of only selected entities. This technique allows the user to "discriminate" these *Entity Types* on the basis of the layer or level. The user must understand that paved roads are to be placed on a different level from unpaved roads.

Other GIS applications, including ArcInfo, do not store data with level/layer assignments, but rather organize graphic entities based on a specific *Entity Type* (road) with a "discriminator" included as a part of the attached attribute data (paved or unpaved). This allows the display of all roads without having to access multiple drawing files. This additional differentiation using the attribute data has been included in this version of the SDSFIE, to allow for differentiation of these *Entity Types* using layer/level or tabular attribute, or even both if desired.

The individual features are logically grouped into *Entity Types* with the inclusion of a discriminator field in the corresponding attribute data. Because this discriminator has only discrete values, it is defined with respect to a domain table, which defines the values which discriminate the *Entities*. Continuing with the previous example, the attribute table associated with Entity Type roads (trvehrd) now contains an attribute which defines the paved status of the road (paved_d) which refers to the domain table (d_pavstt) which contains the values "PAVED" and "UNPAVED". A user or developer can now select the roads which have the paving characteristics desired, or ignore the discriminator completely and display all roads. The technique allows for maximum flexibility in displaying only the information desired to convey the maximum amount of information.

Within the standard, the inclusion of the discriminator concept requires the addition of another *Entity* category. This additional category or grouping of

Entities is defined as *Entity Types* which consist of a given graphic feature "road". These *Entity Types* are normally included in the standard as nouns, while the discriminators represent adjectives which further define or describe these nouns. This modification significantly changes the format of this release of the standard.

Attribute Table. An *Attribute Table* is a relational database table containing data, or information, about a specific SDSFIE entity. SDSFIE *Attribute Tables* are linked directly to a graphic entity (using the electronic tools provided with CADD and GIS software) they are classified as "graphic" (i.e., SDSFIE) attribute tables. FMSFIE attribute tables are indirectly linked to graphic entities via Foreign Key Joins to the SDSFIE attribute tables.

A database can be defined as a structured collection of data items about a specific topic. A database table can be defined as a group of similar records. A database table is like a spreadsheet where the columns represent the fields, or 'attributes,' and the rows represent the records, such that each row will be associated with a single record. A typical GIS links the graphical element, how it is displayed on the screen, with the associated record in the data table.

The SDSFIE/FMSFIE has been designed for use with relational database management system (RDBMS) software. RDBMS software provide a means of managing the related data contained in one or more database tables. Examples of RDBMS software include Oracle (Oracle Corporation) and Access (Microsoft Corporation), SQLServer, and Informix. RDBMS software provides electronic tools for defining relationships (i.e., connections) between the different database tables. These relationships can be defined as: (1) One to Many (most common); (2) One to One (rare, usually merge tables to one); and (3) Many to Many (needs a junction table). The following figures provide a visual representation of RDBMS terminology and relationships.

The name code of each *Attribute Table* is composed of eight characters, due to a design requirement to support both DOS and Windows 3.1 based GIS and database (e.g., dBase) programs. The first two characters, and next three characters in the table name, reference the parent *Entity Set*, and *Entity Class*, respectively. The last three characters in the table name represent a particular Attribute Table.

Some equivalent terms for *Attribute Table* used in GIS and relational database management system (RDBMS) software include:

"feature attribute table" - in MGE.

"attribute table" - in ArcInfo.

"database table" - in RDBMS software.

Domain Table. Domain tables contain standardized lists of permissible values for specific attributes. They provide a predefined finite set of allowable values, which may be enlarged by each user. Included are diverse tables of units of measure, types, styles, status, names, methods, materials, dispositions, sources, dimensions, data, classes, building numbers, etc. The user can add to these lists and range domains installation-specific values as needed.

Two categories of Domain Tables are included in the SDSFIE: (1) List Domains which provide a "picklist" of allowable discrete values, and (2) Range Domains which provide a range (i.e., the minimum and maximum) of allowable discrete values.

Each domain table name code is restricted to eight characters with the first two characters being "d_". All attributes whose value is constrained by a domain value have a name code which ends in "_d". The name code for all attributes whose values are defined by the "unit of measure" Domain Table (d_uom) end in "_u_d".